

Multilayer Image Steganography based on DCT Transform and Mod16 Algorithm

¹Venkata Satish Babu. B, ²Sri CH. Ratna Babu

^{1,2}Dept. of CSE, R.V.R & J.C College of Engineering, Guntur

Abstract

The combination of steganography and cryptography provides privacy in communication and security to confidential information. This paper presents a multi-layer steganography based on 2D-DCT and Mod16 Algorithm. 2D-DCT based steganography using mask matrix is performed at level1 and initially at level2 an encryption algorithm is applied and then Mod16 algorithm is applied to obtain final stego image. The process of extracting the confidential information is the reverse process to the previously done. While maintaining the image imperceptible, the Mod16 algorithm has achieved greater embedding capacity and additional security. Experimental results demonstrate that proposed system is resistant to steganalysis methods like χ^2 attack [7].

Keywords

Mask Matrix, Mod16 Embedding, Image Contrast, Multilayer Steganography, 2D-DCT

I. Introduction

Steganography is the art of concealing secret information. In short sentence it is a 'covered hidden writing'. It is different concept from cryptography. Cryptography provides security to secret information but it doesn't hide the communication between the parties, whereas steganography provides such privacy between communicating parties. Together they can provide stronger security and privacy.

General process of every steganographic method is to select cover image (image used to hide information), and a particular method of hiding is used to hide information in the cover image which finally produces the stego image (image that contain secret information) as a result. Further this process can be reversed (or) may adopt some other method to obtain hidden secret information. This hidden information can be in any format like image, text, audio, video etc.

Many methods have been proposed over the years. The main goal of all these methods is to maintain two important benchmarks, one is capacity and another one is imperceptible in nature to other parties not involved in communication, here capacity refers to amount of information that we are able to hide in the cover image. Some methods like "ConDith" [2] calculate the difference between a pixel and its neighbors. If the difference exceeds a minimum threshold then pixel can be used to embed a message bit. "ConText" [2] method divides the image in non overlapping block of size 3x3 pixels. It again divides each 3x3 blocks in to four 2x2 sub-blocks and selects the central pixel from them if there is three different gray scales in each of the 2x2 sub block. Both of these methods use LSB (Least Significant Bit) method for embedding which is not immune with visual attacks, compression, and less capacity. "Mod4 embedding method" based on image contrast uses shortest route modification scheme to reduce the pixel modifications. It hides 2bits in each 2x2 non-overlapping spatially adjacent pixels.

In general, areas with high contrast and noise like regions are selected for embedding to avoid distortion. Modifications to low contrast regions will be perceptible to the human eye easily

[1]. "Mod4 embedding algorithm" [4] selects high contrast areas for embedding. "Mod16 embedding algorithm" is the extension of "Mod4 embedding algorithm" which hides 4 bits per pixel. Another embedding method is '2D-DCT' [3] which uses low frequencies to represent the secret image and use mask matrix that selects mid frequencies of cover image for embedding the low frequencies of secret image.

Using '2D-DCT' and 'Mod16' algorithms at multiple levels provide additional security than individual 'Mod4' algorithm and DCT based method. The result obtained at one level is passed as an input to another level.

In this paper, we first of all discuss about overall process of proposed method in detail. After we discuss about 2D-DCT based embedding method using mask matrix and in next section we will discuss about Mod16 embedding algorithm. Later we will see the experiment results obtained by the proposed method.

II. Multilayer Steganography

Multilayer steganography using DCT transform and Mod16 algorithm is performed at two different levels. In level1 2D-DCT based method is used and at level2 Mod16 algorithm is used.

At level1 cover image (CV1) and secret image (SEC1) is taken as an inputs and stego image (STG1) is obtained as output. SEC1 is embedded in CV1 using 2D-DCT. At level2 we have two different inputs one is another cover image (CV2) and stego image (STG1) obtained at level1. STG1 is embedded in CV2 using Mod16 algorithm and the output is stego image (STG2). Secret image (SEC1) given at level1 is obtained by reversing the entire process.

If size of secret image more than cover image, the process must be aborted, otherwise any compression algorithm can be used. The overall process of this multilayer steganography is given in fig. 1.

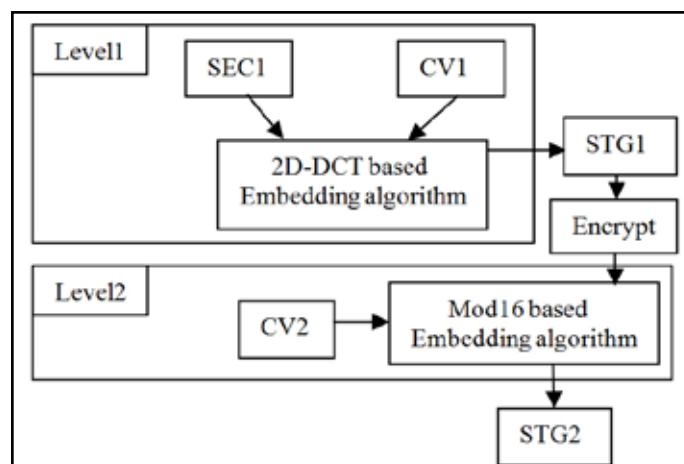


Fig. 1: Multilayer Steganography

III. DCT Transform

2D-DCT (Discrete Cosine Transform) [3] is used to convert image in spatial domain to frequency domain and vice-versa. DCT is similar to DFT (Discrete Fourier Transform) but it uses only real numbers. In DCT we use low frequencies because it contains

more image information than higher ones.

The reason for using 2D-DCT is, because it is involved with fewer computations, its periodicity is continuous and it uses only the cosine waves which are much more efficient. 2D-DCT is performed at level1. The entire process of level1 is given as

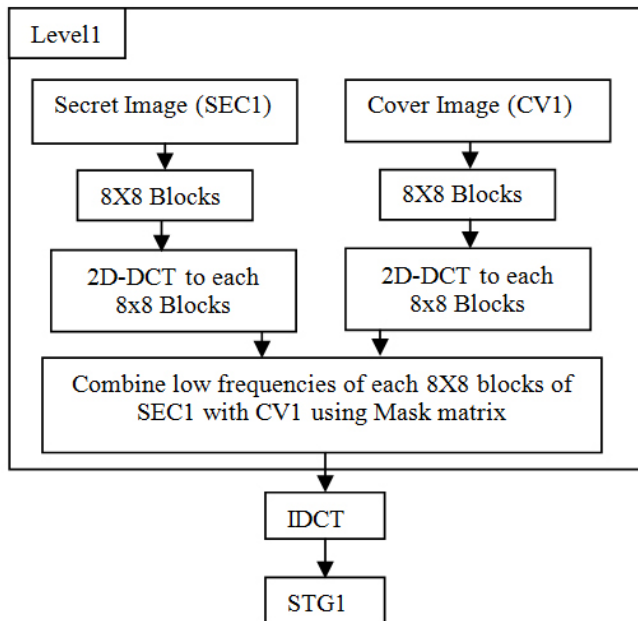


Fig. 2: Level1 Hiding

For given image $f(x,y)$ with the image of size $M \times N$ 2D-DCT is given as

$F(u,v) =$

$$\alpha(u)\beta(v) \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} f(x,y) \cos \frac{(2x+1)u\pi}{2M} \cos \frac{(2y+1)v\pi}{2N} \quad (1)$$

The inverse transform is

$f(x,y) =$

$$\sum_{k=0}^{M-1} \sum_{l=0}^{N-1} \alpha(u)\beta(v) F(u,v) \cos \frac{(2x+1)u\pi}{2M} \cos \frac{(2y+1)v\pi}{2N} \quad (2)$$

Where $x, u=0, 1, \dots, M-1; y, v=0, 1, \dots, N-1$

$$\alpha(u) = \frac{1}{\sqrt{M}} \text{ for } u = 0, \sqrt{2/M} \text{ for } u = 1, \dots, M-1,$$

$$\beta(v) = \frac{1}{\sqrt{N}}, \text{ for } v = 0, \sqrt{2/N}, \text{ for } v = 1, \dots, N-1,$$

When we divide secret image (SEC1) in to 8x8 sub blocks and after applying DCT transform to each individual 8x8 sub block, a very few frequencies (DCT coefficients) located in upper left corner in each 8x8 sub block is enough to represent the entire image. Those few low frequencies enough to retain the secret image (SEC1).

$$Y = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Similarly we divide cover image (CV1) at level1 to 8x8 sub blocks and apply DCT transform to each 8x8 sub block. After performing transformation we can identify the locations in each 8x8 transformed sub block to embed the SEC1 DCT coefficients. This is done by using the mask matrix [3] which identifies the locations to hide. Example of mask matrix is given as

$$\text{Mask matrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

A. Hiding Algorithm Based on Mask Matrix

- Divide the cover image (CV1) and secret image (SEC1) in to 8x8 sub blocks and apply DCT transform to each individual sub block.
- Identify embedding locations in each 8x8 sub block of CV1 using mask matrix. If secret image DCT coefficients less than the available locations in CV1 pad them with zeros and modify the DCT coefficients as equation (3). Apply inverse transform to obtain stego image (STG1).

$$DC'_i = DC_i + x * f_i, \quad i=1 \dots Nu(L) \quad (3)$$

DC = DCT coefficient of cover image.

DC' = new DCT coefficient of stego image.

f = DCT coefficient of secret image

x = intensity factor (taken $x=0.03$).

$Nu(L)$ = number of locations taken in mask matrix

- To obtain secret image embedded, apply DCT transform to each 8x8 block of stego image (STG1). Use mask matrix to find locations in each 8x8 block of transformed matrix and obtain the DCT coefficient of secret image use the formula.
- After calculating and arranging all coefficients in order, apply inverse transform to get back the secret image (SEC1). Here steps (3) and (4) are performed at the end of level2 (after Mod16 algorithm).

IV. Mod 16 Embedding Algorithm

Mod4 embedding algorithm which divides an image in to 2x2 blocks and selects valid 2x2 blocks based on threshold (image contrast) for embedding. Single pixel with high contrast in every 2x2 valid block is selected and modified according to 'shortest route scheme' which is intended to reduce amount of modifications to pixels. Mod4 algorithm embed 2 bits for one 2x2 valid block. Mod16 embedding algorithm is the extension of Mod4 embedding algorithm. Mod16 algorithm is capable of hiding 16 bits of one 2x2 valid block. Similar implementation as Mod16 is Mod8 algorithm which embeds 12 bits for one 2x2 valid Block. Entire process of Mod16 algorithm performing at level2 is given as fig. 3

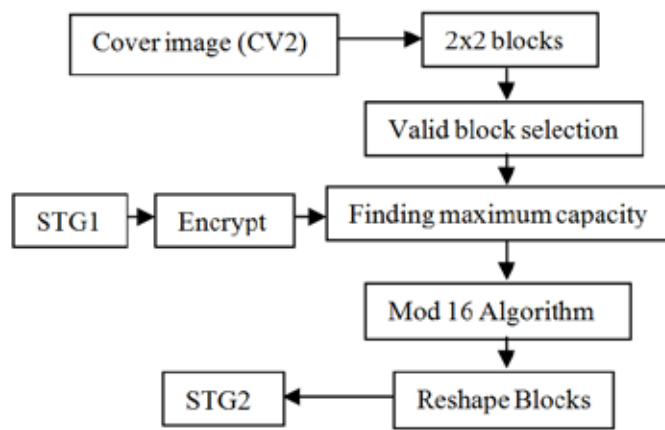


Fig. 3: Level 2 Hiding

Before implementing the Mod 16 algorithm we have to perform simple encryption algorithm to stego image obtained at level1 hiding. RSA algorithm is implemented to encrypt STG1 and converted in to sequence bits.

A. Valid blocks selection

At first CV2 is divided in to 2x2 Blocks. The 2x2 Block is selected as valid block if the average difference between the gray level values of the pixels of that and it's mean (C) exceeds a threshold (minimum contrast) [1], as describe in equation (5)

$$\text{Valid block (Q): } C = (1/4 * \sum |x - m_Q|) > T \quad (5)$$

Where m_Q is the mean gray level value of the pixels in the valid block (Q) and T is the minimum contrast defined by the user. A different value of threshold is taken in this work. Mod4 algorithm leaves 3 pixels useless even if it is valid block. Whereas Mod16 will make use remaining 3 pixel values also. Each pixel in 2x2 valid block is used to embed 4bits of STG1.

Mod16 algorithm calculates the mod value of each and every pixel value of valid block given as

$$\delta(\sigma_Q, 16) = \sigma_Q \bmod 16 \quad (6)$$

σ_Q -It is a single pixel value(intensity value) [4] in valid block(Q). The possible values are between {0, 1, 2, 3,...,15}

Where m_Q is the mean gray level value of the pixels in valid Block (Q) and T is the minimum contrast defined by the user. A different threshold is taken in this work. Mod4 algorithm leaves 3 pixels useless even if it is valid block. Whereas Mod16 will make use remaining 3 pixel values also. Each pixel in 2x2 valid block is used to embed 4bits of STG1.

B. Mod16 embedding Algorithm

The inputs to Mod16 algorithm is STG1 (stego image of level1) and CV2 (cover image of Level2). The embedding process is done in several steps

1. If size encrypted STG1 greater than embedding capacity of CV2, then embedding process halts and compression algorithm can be used.
2. STG1 is encrypted and arranged in to sequence of bits. And $\delta(\sigma_Q, 16)$ of each pixel in valid block is calculated. After calculating $\delta(\sigma_Q, 16)$, we consider the 4bits of encrypted STG1 at a time and converted in decimal format(d). Then modify $\delta(\sigma_Q, 16)$, such that ' $\delta(\sigma_Q, 16) = d$ '. The possible values of d is in between the range {0,1,2,3,...,15}. These modifications can be done using modification table given below (Table 1). This modifications mainly contains {+1,+2,+3,+4,+5,+6,+7,+8,-1,-2,-3,-4,-5,-6,-7,-8} values. These values are added are

subtracted from the ' σ_Q ' to make ' $\delta(\sigma_Q, 16) = d$ ' possible. All this process is continued by making $\delta(\sigma_{Qi}, 16) = d_i, i=1,2,3,...,n$, and this operation is performed until we finish the bits of STG1

3. Perhaps by adding or subtracting large values like +8,+7,+6,+5,+4,+3 (or) -8,-7,-6,-5,-4,-3 will distort the image and reduce number of valid blocks during extraction process, to avoid this problem we can divide above positive values with 100 and negative values with 1000 and add or subtract those values to $\delta(\sigma_Q, 16)$, by doing this we can achieve PSNR ratio above 80. And while extracting we simply multiply with 100&1000 to positive and negative values.
4. After modification, the result is stego image (STG2). To extract exact STG1 in encrypted form, all the process performed until now has to be reversed

Table 1: Modification Table

	0	1	2	3	4	5	6	7
0	--	-1	-2	-3	-4	-5	-6	-7
1	+1	--	-1	-2	-3	-4	-5	-6
2	+2	+1	--	-1	-2	-3	-4	-5
3	+3	+2	+1	--	-1	-2	-3	-4
4	+4	+3	+2	+1	--	-1	-2	-3
5	+5	+4	+3	+2	+1	--	-1	-2
6	+6	+5	+4	+3	+2	+1	--	-1
7	+7	+6	+5	+4	+3	+2	+1	--
8	+8	+7	+6	+5	+4	+3	+2	+1
9	-7	+8	+7	+6	+5	+4	+3	+2
10	-6	-7	+8	+7	+6	+5	+4	+3
11	-5	-6	-7	+8	+7	+6	+5	+4
12	-4	-5	-6	-7	+8	+7	+6	+5
13	-3	-4	-5	-6	-7	+8	+7	+6
14	-2	-3	-4	-5	-6	-7	+8	+7
15	-1	-2	-3	-4	-5	-6	-7	+8
	8	9	10	11	12	13	14	15
0	+8	+7	+6	+5	+4	+3	+2	+1
1	-7	+8	+7	+6	+5	+4	+3	+2
2	-6	-7	+8	+7	+6	+5	+4	+3
3	-5	-6	-7	+8	+7	+6	+5	+4
4	-4	-5	-6	-7	+8	+7	+6	+5
5	-3	-4	-5	-6	-7	+8	+7	+6
6	-2	-3	-4	-5	-6	-7	+8	+7
7	-1	-2	-3	-4	-5	-6	-7	+8
8	--	-1	-2	-3	-4	-5	-6	-7
9	+1	--	-1	-2	-3	-4	-5	-6
10	+2	+1	--	-1	-2	-3	-4	-5
11	+3	+2	+1	--	-1	-2	-3	-4
12	+4	+3	+2	+1	--	-1	-2	-3
13	+5	+4	+3	+2	+1	--	-1	-2
14	+6	+5	+4	+3	+2	+1	--	-1
15	+7	+6	+5	+4	+3	+2	+1	--

Rows=d

Columns= $\delta(\sigma_Q, 16)$ values

Columns represents the 'd' values resulted from binary to decimal conversion of 4 bits of stego image(STG1) taken at time ranges from {0,1,2,...,15}. And row represent the ' $\delta(\sigma_Q, 16)$ ' value of each pixel. Symbol '--' in table represent when ' $\delta(\sigma_Q, 16) = d$ ' i.e. no modifications to $\delta(\sigma_Q, 16)$.

We can represent this entire table in simple formula given as

$$X = d - \delta(\sigma_Q, 16) \quad (7)$$

$$\sigma_Q' = \sigma_Q + X;$$

X=value to be add to σQ
 σQ =pixel intensity value
 $\sigma Q'$ = pixel updated value

For example embedding bit is 0001($d=1$) and if pixel value(σQ) is 162 then $\delta(\sigma Q, 16)=2$, now we have to make ' $\delta(\sigma Q, 16)=d$ ' to embed the $d=1$ for this we have to subtract '1' to σQ according to the table given above(1,2)='-'1' (i.e. is $\sigma Q = \sigma Q - 1$) so now ' $\delta(\sigma Q, 16)=d$ '. Similar operation can be performed to each and every pixel of valid block until all STG1 encrypted bits are finished. We can easily update the pixel value using the equation (7)

After doing all modifications to CV2 (cover image) at level2 we will get stego image (STG2) level-2. Obtaining STG1 from STG2 is reverse process to the previously done. After obtaining STG1 we can perform steps (3) and (4) in section III to get SEC1.

Another simple example of total process at level1 and level2 is given as

1. After performing step (1) and (2) in section III we get STG1. Applying RSA to STG1 and convert it in to sequence of bits.
2. let us consider first 16 bits '0111 0001 1100 1001' and a valid block $Q=[160, 162; 161, 163]$. First 4 bits decimal value is $d=7$ and first pixel value(σQ_1) in valid block is 160 then $\delta(160, 16)=0$. we have to make 160 as 167 because doing mod16 operation on 167 gives 7 which is equal to d i.e. $\delta(160+7, 16)=7$. Since we divides ' d ' value with 100 for positive value and 1000 for negative values, σQ_1 would be 160.07.

Next value of valid block ' Q ' is 162(σQ_1) and $d=1$ (0001) and $\delta(162, 16)=2$. According to 'Table I', we have to subtract 1 from 162. Since negative values are divides by 1000, we subtract 0.001 from 162 results 161.9990. Doing same operation to the remaining two elements. The resulted updated valid block(Q) is given as $Q = [160.07, 161.9990; 164.9950, 163.06]$

This updated valid block is reinserted in its place. Doing this operation to all valid blocks and reinserting will result the stego image (STG2).

3. In level2 extraction phase retrieve the valid blocks again. Consider the valid block (Q) reinserted during level2 hiding phase.

$Q = [160.07, 161.9990; 164.9950, 163.06]$

Consider $\sigma Q_1=160.07$. Obtaining encrypted bits of steganographic image (STG1) is performed in the following manner.

$t = \sigma Q_1 - \text{floor}(\sigma Q_1) \dots (0.07)$

If t greater than 0.01

$d = t * 100 \dots (d=7(0111))$

else /*in case of 161.9990*/

$t = \text{ceil}(\sigma Q_1) - \sigma Q_1 \dots (0.0010)$

$d = d * 1000 \dots (d=1(0001))$

In this way, we perform extraction process to remaining valid blocks to obtain ' d ' values and then convert them in to sequence bits and apply RSA decryption to obtain STG1.

4. Perform the steps (3) & (4) in Section III to obtain SEC1.

Another important point is to remember '-8' is not included because result by adding +8 to ' σQ ' is equal to adding -8 to ' σQ '.

V. Experiment Results

The proposed multilayer steganography is carried out on variety of images in MATLAB software. Images are first converted in to type 'double' to perform Mod16 algorithm. In order to know the efficiency of the proposed algorithm, we compare the hiding capacity of the proposed algorithm with previous algorithms, and

'PSNR' ratio is calculated, χ^2 analysis is performed.

A. Peak signal to Noise Ratio

PSNR [6] is used to evaluate the quality of the stego image by comparing it with cover image. It measures the percentage of stego data with cover image percentage and evaluates the quality of stego image after hiding the secret information. Let us consider the cover image $C(i, j)$ that contains an $N \times N$ pixels and stego image $S(i, j)$ contain $N \times N$ pixels. Mean squared error of stego image as follows:

$$MSE = 1/[N * N]^2 \sum_{i=1}^N \sum_{j=1}^N [C(i, j) - S(i, j)]^2 \quad (8)$$

The PSNR is calculated using formula

$$PSNR = 10 \log_{10} 255^2 / MSE \quad (9)$$

Different types of standard images are taken at two different levels. Table 2 shows the PSNR values of cover and stego images obtained at level1. Table 3 shows the PSNR values of cover image and stego images obtained at level2.



Fig. 4: Cover Images (192x192) Taken at Level1 at a Intensity Factor(x)=0.03

Table 2: PSNR of Images at Level1 Hiding

Images	PSNR values
Baboon	43.0506
Land	42.8493
Bowl	39.5075
Barba	38.7377

At level2 following images of dimensions 512x512 are taken at level2 as CV2(cover image)



Fig. 5: Cover Images Taken at Level2

Table 3: PSNR Values of Cover Image at level2

Image	PSNR	Threshold value(T)
jet	81.3113	168.55
Pepper	82.3293	118.1
House	81.2421	168.55
M.K.Gandhi	81.6319	168.55

We can set the threshold value based on size of STG1. All the images have to convert in to gray before all operations.

B. Histogram Analysis

The histograms taken of the original image (or) cover image and stego image of 'Land.bmp' at level1 which is embedded in 'house.bmp' at level2 is shown in fig. 6 & fig. 7.

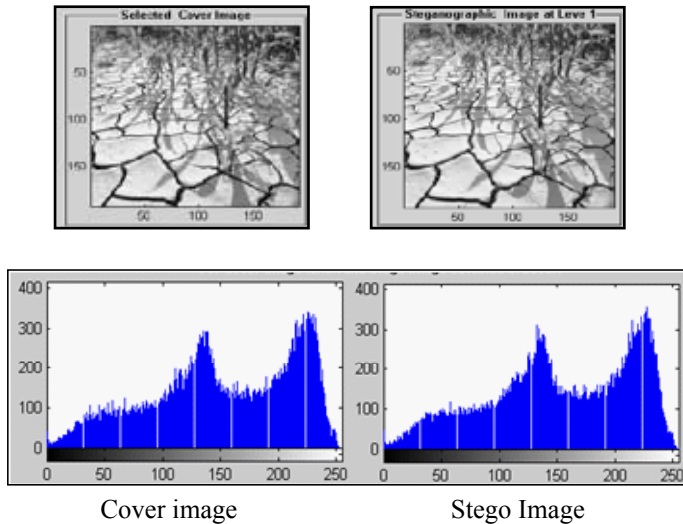


Fig. 6: Histograms of Stego and Cover Image at Level1

It clearly shows both histograms of cover and stego images are identical. It is mainly due to the intensity factor ($\alpha=0.03$). The histograms of cover and stego image of 'house.bmp' at threshold (T) = 118.1 is shown in fig. 7

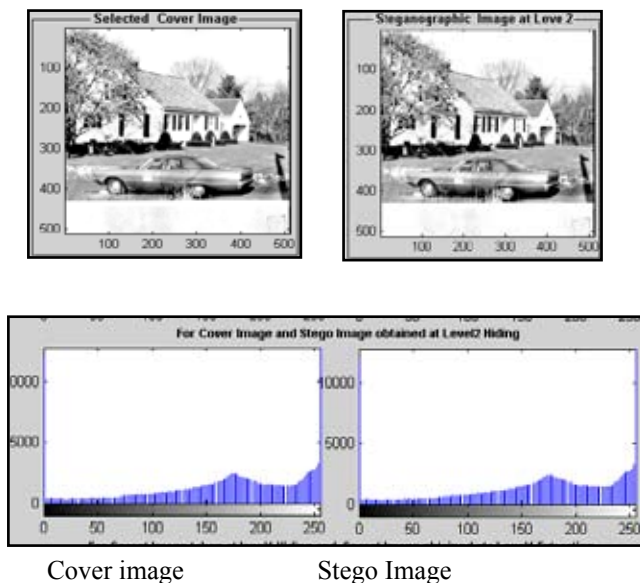


Fig. 7: Histograms of Stego and Cover Image at Level 2

Above histogram shows the effectiveness of Mod16 algorithm used at level2

C. χ^2 Statistical Analysis

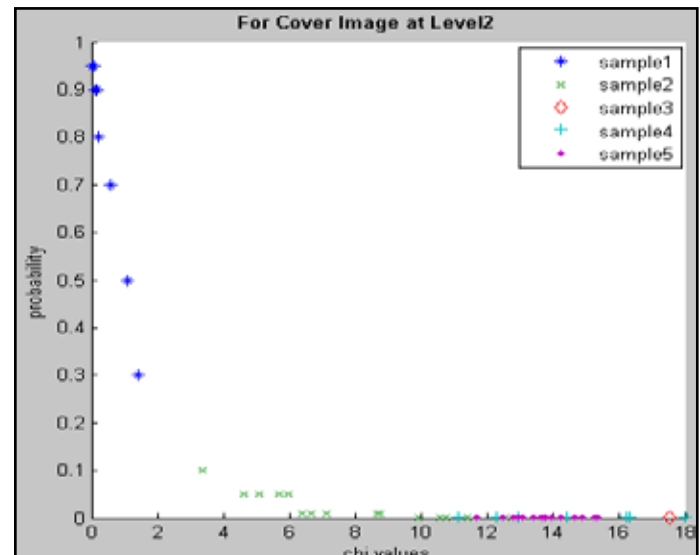
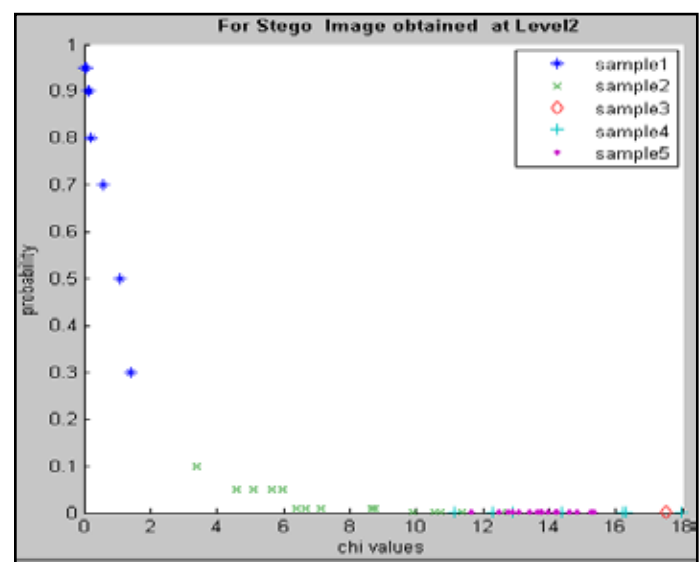
Chi-square (χ^2) [8] is a statistical test commonly used to compare observed data with data we would expect to obtain according to a specific hypothesis. The chi-square test is always tests the 'null hypothesis', which states that there is no significant difference between the expected and observed result. The formula for calculating chi-square (χ^2) is:

$$\chi^2 = \sum (\text{observed} - \text{expected})^2 / \text{expected} \quad (10)$$

In χ^2 statistical analysis we have mainly a critical value table [7] with number of critical values. In this table columns represent the probability and rows represent degrees of freedom i.e. $df = (\text{number of outcomes}) - 1$. If we perform analysis on 2×2 sample distribution then $df = (2 - 1) = 1$. Most of the time probability value (α) would be 0.05. So ' df ', α is used to look up critical value table, the value of table must be 3.841. If our calculated ' χ^2 ' value of 2×2

sample greater than 3.841 then 'null hypothesis is rejected i.e. there is significant difference between observed and expected value else 'null hypothesis' is to be accepted.

Here we have taken five 8×8 samples from cover image and five 8×8 samples stego image in same locations. χ^2 statistical analysis is performed on each individual 8×8 block of stego image and on each individual 8×8 block of cover image separately by dividing it in to 2×2 blocks. Later χ^2 values and their respective probability for each 2×2 blocks in 8×8 block in both cover image and stego image is calculated. After calculation a scatter graph is drawn between probability and χ^2 values of each 5 samples of stego and 5 samples of cover separately. Scatter graph of χ^2 analysis is shown in fig. 8 & fig. 9

Fig. 8 χ^2 analysis of 'house.bmp' cover image at level2Fig. 9: χ^2 Analysis of 'house.bmp' Stego Image at Level2

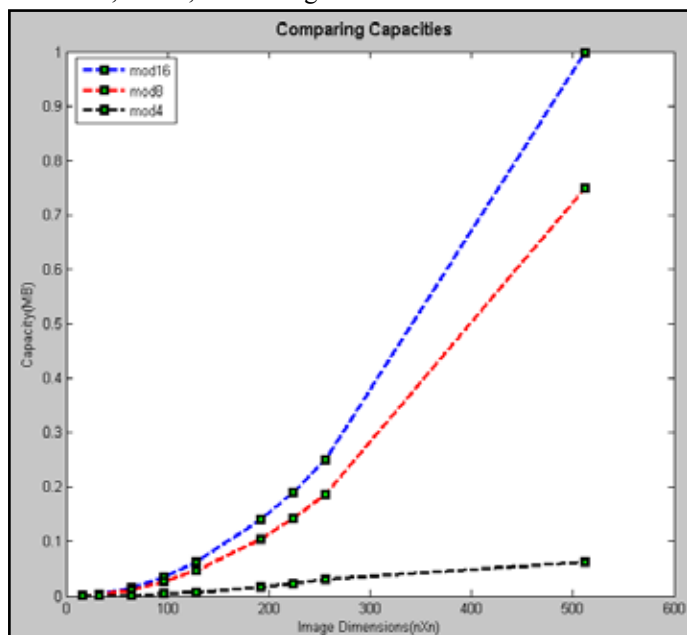
We can observe that there is no any difference between Fig.8 and Fig.9. Thus χ^2 statistical analysis is failed to confirm that stego image contain secret data.

D. Comparing the Capacities

Coming to the embedding capacity of Mod4 algorithm is 2 bits for 2×2 block. Mod4 algorithm leaves remaining three pixel values useless and minimizes the embedding capacity. Coming to the embedding capacity of Mod16 algorithm utilizes the four pixel

values in 2x2 block and embed 16 bits per one 2x2 block and increases the embedding capacity.

The graph (Fig.10) below compares the embedding capacity of the Mod4, Mod8, Mod16 algorithms



$n = \{16, 32, 64, 128, 192, 224, 256, 512\}$

Fig. 10: Comparing Capacities of Algorithms Mod4, Mod8, Mod16

The process of Mod8 algorithm is similar to the Mod16 algorithm, but it hides 12 bits per one 2x2 valid block. Above graph is drawn between the 'Capacity' and 'Image Dimensions (nXn)'. We can observe that for 512x512 image Mod16 algorithm has reached capacity to store '1 Mega Byte' of information by using every pixel. Where Mod4 capacity stays below the 100Kb.

VI. Conclusion

In this paper, a multilayer steganography method based on DCT transform and Mod16 algorithm is proposed to provide better security level and to increase embedding capacity than previous algorithm. Proposed algorithm is also withstand to statistical attack like χ^2 analysis with acceptable PSNR. Implementing DCT based transform on level1 and Mod16 in one level along with RSA algorithm provides more security. Our future work includes Refining the pixel selection method for embedding and to implement Mod-K algorithm to embed even more capacity than Mod16 algorithm.

Implementing steganography more than 2 levels for providing even more security than present algorithm.

References

- [1] Mah Mahdi Ramezani and Shahrokh Ghaemmaghami, "Adaptive Image Steganography with Mod-4 Embedding using Image Contrast", in proceedings of 7th IEEE Conference on Consumer Communications
- [2] R. H. M. Dulce, R. C. Raul, F. U. Claudia, "Adaptive Steganography based on textures", in Proceedings of the 17th International Conference on Electronics, Communications and Computers, pp. 34-39, Sep 2007.
- [3] Sun Xijing, Si Shoukui, Tian Shurong, "The Image Hiding Algorithm Based on the Mask Matrix", 2011 IEEE

- [4] K. Pramitha, Dr. L. Padma Suresh, K.L. Shunmuganathan "Image Steganography using Mod-4 Embedding Algorithm based on Image contrast", Proceedings of 2011 International Conference on Signal Processing, Communication, Computing and Networking Technologies (ICSCCN 2011).
- [5] X. Qi, K. Wong, "An Adaptive DCT-Based MOD-4 Steganographic Method", in International Conference on Image Processing (ICIP'05).
- [6] Peak Signal to Noise Ratio. Wiki Wikipedia network, [Online] Available: <http://zh.wikipedia.org/wiki/%E5%B3%B0%E5%80%BC%E4%BF%A1%E5%99%AA%E6%AF%94>.
- [7] [Online] Available: http://psychology.ucdavis.edu/sommerb/sommerdemo/stat_inf/tutorials/chisqhand.htm.
- [8] Andreas Westfeld, Andreas Pfitzmann, M. Wegmüller, "Attacks on Steganographic Systems Breaking the Steganographic Utilities EzStego, Jsteg, Steganos, and S-Tools".



Venkata Satish Babu has completed his BTECH degree in CSE from 'Acharya Nagarjuna University' in 2010. He is currently pursuing his MTECH degree in Computer Science from RVR & JC college of Engineering, Guntur, AP, India. His research interests are 'cryptography and network security', 'Steganography'.



Sri. Ch. Ratna Babu has completed his MTECH from 'Acharya Nagarjuna University' in 2006. He is currently registered for Ph.D. at Acharya Nagarjuna University, Guntur, AP, India. He has published the journal named "A Randomized Secure Data Hiding Algorithm Using File Hybridization For Information Security" in International Journal on computer science and engineering (IJCSSE). He has written a book on "C-Book for distance Education ANU", BPB Publication. His research interests include 'Cryptography and Network Security'. He is currently working as an Assistant Professor in Computer Science Department, RVR and JC College of Engineering, Guntur, Andhra Pradesh, India.